REPORT DOCUMENTATION PAGE

OMB NO. 0704-0188

1. AGENCY USE ONLY (Leave Blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
	05/29/01	Final Report 04/15/98-04/14/01

4. TITLE AND SUBTITLE

Reliability of Large Scale Disk Arrays

5. FUNDING NUMBERS

C-DAAG-559810272

6. AUTHOR(S)

Charles J. Colbourn

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)

University of Vermont

8. PERFORMING ORGANIZATION REPORT NUMBER

None

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)

U. S. Army Research Office
P.O. Box 12211
Research Triangle Park, NC 27709-2211

10. SPONSORING / MONITORING AGENCY REPORT NUMBER

37846.36-CI-DPS

11. SUPPLEMENTARY NOTES

The views, opinions and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy or decision, unless so designated by other documentation.

12 a. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release: distribution unlimited.

12 b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Performance and reliability are major concerns in the design of large disk arrays. Hellerstein et al. pioneered the study of erasure-resilient codes that allow one to reconstruct data without loss in the presence of disk failures. Chee, Colbourn, and Ling used the close connection between erasure-resilient codes and certain combinatorial designs to establish much improved asymptotic and exact existence results for these codes. The design-theoretic approach provided the scientific basis for the project.

In the subsequent sections, we first provide the relevant background on the design of erasure codes for RAID, contrasting with the more extensively studied erasure codes for digital communications. Then we summarize highlights of the research in the ARO project now completed. Our research effort on codes for disk arrays revealed an unexpected means of optimizing I/O performance through appropriate orderings of codewords. Indeed our simulation results show a marked improvement in performance when codewords are ordered such that consecutive sets of codewords exhibit a maximum overlap. We undertook an investigation of optimal orderings for triple erasure codes, and obtained substantial results on orderings for double erasure codes.

14. SUBJECT TERMS

Disk Array, RAID array, reliability, combinatorial design

15. NUMBER OF PAGES

16

16. PRICE CODE

17. SECURITY CLASSIFICATION OR REPORT	18. SECURITY CLASSIFICATION ON THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UL

NSN 7540-01-280-5500

Standard Form 298 (Rev.2-89)
Prescribed by ANSI Std. 239-18
298-102

Enclosure 1

DAAG 559810272 Final Report

"Reliability of Large Scale Disk Arrays"

Charles J. Colbourn, Computer Science
The University of Vermont
Burlington, Vermont

May 30, 2001

1. Foreword

Performance and reliability are major concerns in the design of large disk arrays. Hellerstein et al. pioneered the study of erasure-resilient codes that allow one to reconstruct data without loss in the presence of disk failures. Chee, Colbourn, and Ling used the close connection between erasure-resilient codes and certain combinatorial designs to establish much improved asymptotic and exact existence results for these codes. The design-theoretic approach provided the scientific basis for the project.

In the subsequent sections, we first provide the relevant background on the design of erasure codes for RAID, contrasting these with the more extensively studied erasure codes for digital communications. Then we summarize highlights of the research in the ARO project now completed. Our research effort on codes for disk arrays revealed an unexpected means of optimizing I/O performance through appropriate orderings of codewords. Indeed our simulation results show a marked improvement in performance when codewords are ordered such that consecutive sets of codewords exhibit a maximum overlap. We undertook an investigation of optimal orderings for triple erasure codes, and obtained substantial results on orderings for double erasure codes.

2. Table of Contents

4. Statement of the Problem Studied

Over the last decade, there has been a sustained exponential advance in the density and performance of semiconductor technology. With this progress has come faster microprocessors as well as larger and faster primary memory devices. Improvements in secondary storage systems, on the other hand, have not kept pace. While the performance of RISC microprocessors has been increasing by more than 50% per year [42], disk transfer rates, which depend

on the speed of mechanical movements and magnetic media densities, have only improved by about 20% each year [10]. This phenomenon has transformed many computationally-bound applications into I/O-bound applications.

Amdahl [2] predicted three decades ago that unless accompanied by corresponding increases in secondary storage performance, big increases in microprocessor performance can only bring about marginal improvements in overall system performance. This disparity has led to the consideration of parallelism as a means to speed up secondary storage systems. Several ideas have been proposed as to how parallelism can be exploited. The most important and successful is the *disk array architecture.*

The *disk array architecture* organizes many independent small disks into one large logical disk. Small disks are preferable to large ones because they have a lower cost and consume less power. For improved performance, disk arrays employ the concept of *data striping* [45], which spreads data across multiple disks. This allows both single and multiple I/O requests to be processed in parallel by separate disks, thus improving effective transfer rates. A further advantage of disk striping is uniform load balance.

The more disks we have in a disk array, the higher the performance we obtain. Unfortunately, large disk arrays have low probability of having all disks functional. Failures in disk arrays are often assumed to satisfy the memoryless property, that is, the life expectancy of a disk is dependent only upon the condition that the disk is working now. Under this assumption, the reliability of a disk array is modeled by the exponential distribution [29]. As a consequence, for low disk failure rates, the failure rate of a disk array is directly proportional to the number of disks it contains. Many applications, notably database and transaction processing systems, require both high throughput and high data availability of their storage systems. The most demanding of these applications require continuous operation, which in terms of a storage system requires (i) the ability to satisfy all requests for data even in the presence of disk failures, and (ii) the ability to reconstruct the content of a failed disk onto a replacement disk, thereby restoring itself to a fault-free state. These requirements strongly encourage the introduction of redundancy to tolerate disk failures. Disk arrays which incorporate redundancy have come to be known as *Redundant Arrays of Independent Disks* (RAID).

There are three primary types of disk failures. *Transient errors* arise from noise corruption and are dealt with by repeating the requests. *Media defects* are caused by permanent defects in material, and are detected and masked by the manufacturer. *Catastrophic failures* include head crashes and failures of the disk controller electronics. When a disk suffers a catastrophic failure, its data is rendered unreadable, and is effectively erased. We therefore call such a disk failure *an erasure.* For convenience, we also call a set of k disk failures *a k-erasure.* Error-correcting codes can be used to tolerate erasures. However, components in disk arrays allow us to determine exactly where erasures have occurred. It is possible to take advantage of this additional information to derive codes that are better than those based on error-correcting codes.

Elias [28] apparently was the first to distinguish between erasures and errors, and to develop a model of the erasure channel. Rabin [43] investigated erasure-resilient codes for information dispersal. The intended application in Rabin's work arises when losses are frequent, and there is a relatively small overhead in having a large amount of redundant

data. Alon et al. [1] also studied erasure-resilient codes to combat bursty losses in packet-switched networks. The communications environment in which a substantial fraction of packets are erased (lost) is a practical model of the Internet as currently deployed. For this reason, erasure codes that deal with a large fraction of erasures have been very aggressively studied. A sample of the most relevant papers in this area includes [37, 38, 6, 46]. Coding files for broadcast or transmission permits both large scale loss and high levels of redundancy, and involves typically a very large number of information and redundant (check) packets. In disk arrays, however, one finds a very different application of erasure codes. In that context, erasures are much rarer. More importantly, the sizes of the disk arrays involved is orders of magnitude smaller than the number of packets in a file broadcast such as the Digital Fountain [6]. Hence the parameters of erasure codes of interest are quite different in these two applications, despite similar definitions and objectives.

Hellerstein et al. [32] first proposed the use of erasure-resilient codes for large disk arrays. Chee, Colbourn, and Ling [8] extended their work, establishing an extensive combinatorial framework for their study. By interpreting the coding problem in the context of extremal set theory, we have already obtained new classes of optimal and asymptotically optimal erasure-resilient codes. These codes improve and extend previous results in the literature. Our treatment has also revealed interesting and surprising connections to combinatorial design theory. The mathematical study of erasure codes, especially in the disk array context when the number of erasures is small, lies within the theory of error-correcting codes [40]. In this direction, relevant research concerns low density parity check matrices; see [4, 5, 35, 49], especially [4], in which an application to RAID is discussed.

Hellerstein et al. [32] formulate the construction of erasure codes for disk arrays as follows. A system of linear equations modulo 2 specifies the contents of c *check* or *parity* disks as functions of the contents of n *information* or *data* disks. Often these are represented in matrix form using a *parity check matrix* $H = [P|I]$. H is a $c \times (c+n)$ matrix of 0s and 1s. P is a $c \times n$ matrix with rows indexed by check disks and columns by information disks, so that the (i,j) entry is 1 if and only if information disk j is checked by check disk i. I is a $c \times c$ identity matrix with rows and columns indexed by the check disks.

An $[n,c,k]$-*erasure-resilient code*, or briefly an $[n,c,k]$-*ERC*, consists of an encoding algorithm $\mathcal{E}$ and a decoding algorithm $\mathcal{D}$ with the following properties. Given an n-tuple S of stripes, $\mathcal{E}$ produces an $(n+c)$-tuple or *codeword* $\mathcal{E}(S) = (\mathcal{E}_1(S), \ldots, \mathcal{E}_{n+c}(S))$ of stripes such that for any $I \subseteq \{1, \ldots, n\}$, where $|I| = n + c - k$, the decoding algorithm $\mathcal{D}$ is able to recover S from $(I, \{\mathcal{E}_i(S) \mid i \in I\})$. We often call an $[n,c,k]$-ERC a k-ERC when the parameters n and c are not important in the context.

Erasure correction capability (*reliability*) is completely specified by the parity check matrix: A set of disk failures can be corrected if and only if the corresponding set of columns of H is linearly independent modulo 2 [32]. Aspects of *performance* are also specified by the structure of this matrix. In particular, the column sums of P specify the *update penalties*, reflecting the cost of writing redundant information when a data disk is written. The row sums of H specify the *group sizes*, indicating the cost of reconstructing a failed disk. The ratio of c to n is the *check disk overhead*, the additional cost in terms of number of disks which we incur in order to maintain the redundant information. More precisely, we have:

Check disk overhead: This is the ratio of the number of check disks to information disks.

An $[n, c, k]$-ERC has a check disk overhead of c/n.

Update penalty: This is the number of check disks whose content must be changed when an update is made in the content of a given information disk. We call these disks the *disks associated with the information disk.* If m check disks need to be involved in every write, then the parallelism of the disk array is reduced by a factor of $m+1$. Since parallelism is the reason behind using disk arrays, update penalties should be kept as small as possible. The update penalties of an erasure-resilient code with parity-check matrix $H = [C \mid I]$ are the column sums of C.

Group size: This is the number of disks that must be accessed during the reconstruction of a single failed disk. The cost of reconstruction makes small group size desirable, while for load balancing reasons, uniform group size is desirable. The group sizes of an erasure-resilient code are the row sums of its parity-check matrix.

Since updates of data are usually much more frequent than the reconstruction of data lost in erasures, the update penalties are typically of more concern than the group sizes.

For performance reasons, the erasure-resilient codes studied are assumed to satisfy two conditions:

1. We restrict ourselves to *systematic* codes. An $[n, c, k]$-ERC is *systematic* if $\mathcal{E}_i(S) = S_i$, for $1 \leq i \leq n$, where $S = (S_1, \ldots, S_n)$. The stripes $\mathcal{E}_i(S)$, for $n < i \leq n + c$, are called *checks.* This means that the encoding function leaves the data unmodified on some disks. This property is desirable to avoid read penalties associated with decoding when there are no disk failures.

2. We restrict ourselves to *linear* codes over the field of order 2^L, where L is the bit-size of a stripe. In this case, we interpret a stripe as an L-dimensional vector over the field of order 2, and $\mathcal{E}$ is a linear function. Hence, computations used to encode a stripe are restricted to component-wise modulo two arithmetic, that is, the parity (or 'exclusive or') operation $\oplus$. This restriction ensures that encodings and manipulations can be performed efficiently.

5. Summary of the Most Important Results

To date, our research has concerned reliability of parity check matrices whose update penalties are as small as possible, while still correcting for all sets of d or fewer erasures. We have concentrated on cases when $d = 3$ or 4. Chee *et al.*[8] describe optimal codes for correcting three or more erasures arising from Steiner triple systems. These are codes which provide minimal update penalty, small group size and a reasonable check disk overhead. In addition, anti-Pasch Steiner triple systems are a class of codes that provide higher resilience; see[8].

In order to understand performance within a class of codes offering similar erasure reliability and to understand if the added resilience of anti-Pasch codes affects performance we implemented a computer simulation.

RaidSim

RaidSim [33, 34] is a simulation program written at the University of California at Berkeley [34]. Holland [33] extends it to include declustered parity and online reconstruction. The raidSim program models disk reads and writes and simulates the passage of time. The modified version described in [33], was further modified and used as our starting point for experimentation.

Implementing a 3-erasure code is expensive due to the high update penalty for each write. It is therefore imperative to understand how various factors in designing these codes impact overall performance. Our experiments were designed to gain insight into what these factors might be, and how they might affect the behavior of 3-erasure codes. The first issue in mapping and layout of a disk array is that of modeling reads and writes. In a single error correcting disk array the basic unit is a parity stripe or group. There can be no overlap of disk accesses. One check disk is assigned to each information disk in each group. Two basic types of writes have been described [33]. If more than half of the parity stripe is being written then all of the remaining information disks in the stripe are pre-read, the parity for the stripe is computed and the new data and parity are written [33]. If however, the new data writes to less than half of the parity stripe, a *read-modify-write* is preferred [33]. In this case the requested information disks and their associated check disks are pre-read, the new parity is computed and then the new data and parity is written back out.

In a multiple erasure code, some differences arise. The basic assumption that one check disk is assigned to each data disk is no longer valid. Multiple data disks in the same stripe may use the same check disk. This changes the manner in which a write must be performed. In order to optimize performance it is necessary to avoid the unnecessary reading and writing of a disk more than once within a single parity stripe when check disks coincide during an individual write. Indeed, in a small write, if multiple information disks are involved, it may not be necessary to read and write t check disks for each data disk in a t-erasure code. Although, the minimum update penalty of a t-erasure code remains t [32, 36], the actual penalty may be reduced by changing the order of columns in the parity check matrix. We focus our attention here on the read-modify-write. This is the more interesting type of write, since it is an expensive operation in a triple erasure code and hence provides some potential for improvement.

Every *individual* disk write in a three-erasure configuration could involve up to four reads and four writes. When more than one disk in a stripe is being written, however, the number of extra reads and writes per information disk may decrease due to overlap in check disks. A diagram of how read-modify writes might occur in a triple-erasure code can be seen in Figure 1. It shows a scenario in which data is striped across two disks with one check disk in common. This might happen in a case where the two triples {1,2,3} and {1,4,5} are checking two data disks used in the same write. In this case only five check disks are read and written. (Check disk one only needs to be read and written once, saving two disk accesses.)

Thus the various orderings of the columns of the parity check matrix may reduce the overall number of reads and writes.

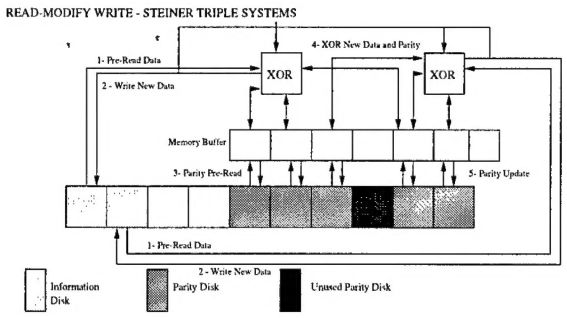

Figure 1: Read Modify Writes in a Steiner Triple System

Simulation Workload			
Percent	**Operation**	**Size(KB)**	**Alignment(KB)**
Read			
100%	Read	72	24
Write			
100%	Write	72	24
Mixed			
82%	Read	72	24
18%	Write	72	24

Table 1: Ordering Workload

Experiments

The performance experiments are run with a workload shown in Table 1. Approximately 100 experiments are run for each system. Each experiment is tested in fault free mode and then with one, two, three, four and five simultaneous failures. All disk failures occur at the start of the experiment and persist for the entire test.

We focus on Steiner triple systems of order 15. There are eighty non-isomorphic systems of order 15. Six different systems were chosen with varying numbers of Pasch configurations. System one and system eighty differ the most in the number of configurations and are therefore the most different structurally. To make sure that we did not neglect other differences in these codes, we chose four other systems.

Three orderings are used for the experiments; these are discussed in more detail in [14]. The first is one in which the overlap among consecutive blocks is maximized for reads or writes that span multiple disks. This could minimize the overall number of disk accesses in a write which spans information disks. The second ordering has the property that every set of three consecutive blocks are pairwise disjoint. The last, designed for comparison, is randomly permuted.

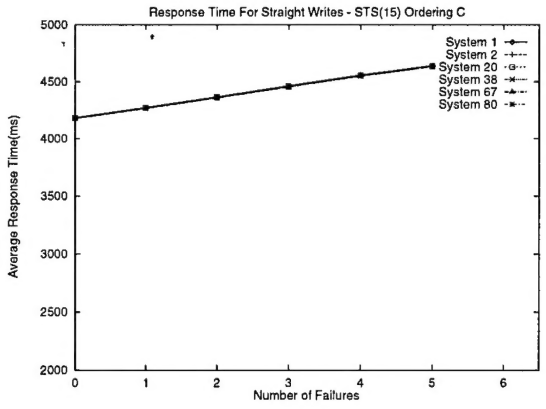

Figure 2: Write Workload Comparison - Ordering C

Performance Results

Interestingly, the structure of the codes does not appear to play a significant role in the performance of these arrays. This is illustrated by Figure 2 which shows the results for the permuted ordering under a workload of straight writes. These simulations do not appear to distinguish between different systems of order 15. Similar results are observed using the other two orderings. The structure of the code, however, does determine the erasure correction capability of a code. Given that no observed performance differences are seen, anti-Pasch Steiner triple systems may be more desirable in situations when higher reliability is desired [8].

Consistently, in all of these experiments, the order of columns in the parity check matrix most reliably predicts performance. The lowest response times appear using the first ordering, labeled A. This is the ordering that attempts to maximize overlap among each group of three consecutive information disks. The second ordering, B, gives the slowest response time. This is the ordering that uses pairwise disjoint consecutive blocks. Response times for the third ordering, C, lie between these two extremes. This is the permuted or random ordering.

In a read-only workload there is no apparent difference in the performance of the various orderings in fault free mode. This is expected since a read workload does not access the check disks (see Figure 3). However, the response time starts to diverge as the number of failed disks increases. This can be attributed to accesses to additional disks required in reconstruction. As seen in Figure 4, the most dramatic difference in performance among the different orderings is seen in a straight write workload. This is expected since writes incur a significant update penalty. Although these account for only a small portion of the accesses in a mixed workload, the overall performance pattern still follows that of a straight write workload. This can be seen in Figure 5.

The results of this experimentation led us to ask fundamental questions about ordering in Steiner triple systems and their application to RAID arrays. However, the more exciting possibility arises for double erasure codes. With current sizes of disk arrays, triple erasure correction is rarely needed [32]. Double erasure correction is needed more often [41], and a

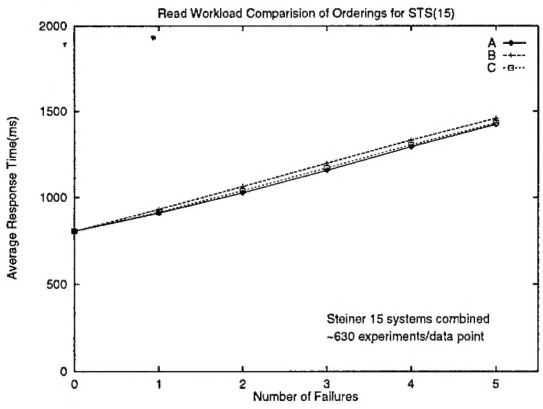

Figure 3: Ordering Results - Straight Read Workload

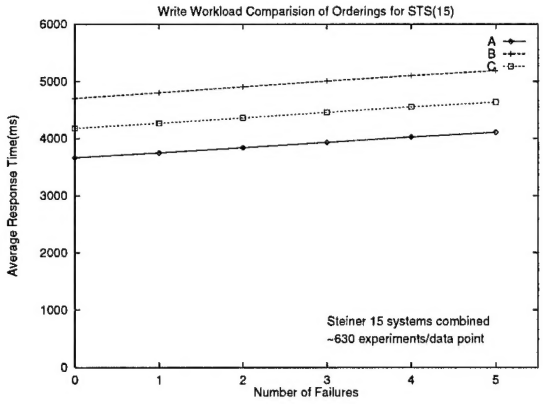

Figure 4: Ordering Results - Straight Write Workload

number of practical schemes have been suggested [4, 41]. The general framework in [8, 32] encompasses the schemes so far studied. In this framework, a scheme for double erasures having all update penalties equal to two can be viewed as a graph. In particular, the P matrix is the incidence matrix of a graph on c vertices and n edges. Then the *reliability* question is to characterize the c-vertex n-edge graph which corrects the largest number of erasures of more than two disks (since all graphs give schemes to correct all sets of two or fewer erasures). The *performance* question is to determine, among those graphs which exhibit acceptable reliability, how to represent the parity check matrix (i.e. how to order it) to optimize user response time, by reducing the effective update penalty for small writes. Both questions are complicated by the need for scalability, the requirement that the disk array be expandable to meet changing needs for information content. Scalability leads to further 'ordering' questions, namely the order in which disks are to be removed from or added to an existing disk array [32].

The impact of optimizing the selection of H for reliability is well understood in principle [32], but the construction of highly reliable codes is not well developed [8]. In fact, these constructions involve difficult open questions on combinatorial designs [30, 36]. The sub-

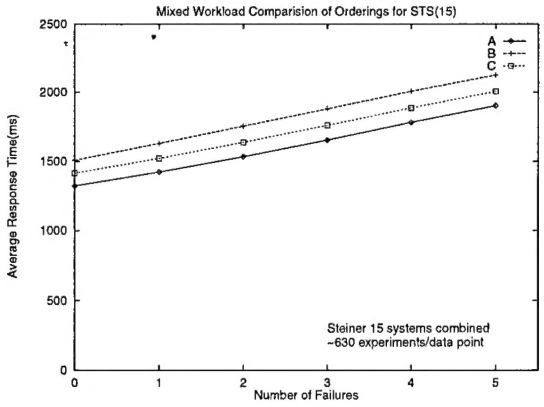

Figure 5: Ordering Results - Mixed Workload

stantial effect of the selection of the ordering of H on performance has become clear [11, 14] as a result of the project research.

In addition to the effort described above which addresses the main focus of the completed grant, the PI has been very active in the areas of combinatorial algorithms, and combinatorial designs. In ARO Project DAAG-98-1-0272, "Reliability of Large Scale Disk Arrays", we have:

1. Implemented a simulation environment for the study of reliability of erasure codes for the correction of three simultaneous erasures [11, 13].

2. Extended computational search techniques for generating optimal codes to correct for three or four erasures [15, 8, 39].

3. Generalized a classical construction technique (the Bose method) for producing triple erasure codes [24, 44].

4. Developed the essential mathematical machinery [36] for a complete solution to Erdös's problem [30] leading to an existence result for triple erasure codes.

5. Obtained preliminary results on structural aspects of erasure codes and their RAID implementation [11, 14].

Concurrently, the methods developed in this research have found broad application to many domains:

1. Multiple access communications for MT-MFSK signaling [26, 50, 51].

2. Design of bus networks [17].

3. Superimposed codes for nonadaptive group testing, for satellite reservation systems [7, 9, 16] and for frameproof codes [47, 48].

4. Quorum systems for mutual exclusion [21, 22].

5. Ring grooming in SONET networks to minimize electrical/optical conversion [25].

General themes underlying these applications are discussed in [20], and specific issues are addressed in [18].

6. List of Publications

1. C.J. Colbourn and A.C.H. Ling, Kirkman triple systems of orders 27, 33, and 39, *Journal of Combinatorial Mathematics and Combinatorial Computing*, to appear (acc Feb01).

2. M.B. Cohen, C.J. Colbourn, L.A. Ives, and A.C.H. Ling, Kirkman triple systems of order 21 with nontrivial automorphism group, *Mathematics of Computation*, to appear (acc Jan01).

3. C.J. Colbourn and P.-J. Wan, "Minimizing drop cost for SONET/WDM Networks with 1/8 wavelength requirements", *Networks* 37 (2001), 107-116.

4. M.B. Cohen and C.J. Colbourn, "Optimal and pessimal orderings of Steiner triple systems in disk arrays", *Theoretical Computer Science*, to appear (acc Aug00).

5. C.J. Colbourn and F. Sagols, "Triangulations and a generalization of Bose's method", *Discrete Mathematics*, to appear (acc Aug00).

6. C.J. Colbourn and A.C.H. Ling, "Quorums from difference covers", *Information Processing Letters* 75 (2000), 9-12.

7. A.C.H. Ling and C.J. Colbourn, "(M,S)-optimal designs with block size three", *Australasian Journal of Combinatorics*, to appear (acc Jul00).

8. F. Sagols and C.J. Colbourn, "NS1D0 sequences and anti-Pasch Steiner triple systems", *Ars Combinatoria*, to appear (acc Mar00).

9. C.J. Colbourn, J.H. Dinitz, and D.R. Stinson, "Quorum systems constructed from combinatorial designs", *Information and Computation*, to appear (acc Nov99).

10. A.A. Bruen and C.J. Colbourn, "Transversal designs in classical planes and spaces", *J. Combinatorial Theory (A)* 92 (2000), 88-94.

11. A.C.H. Ling and C.J. Colbourn, "Modified group divisible designs with block size four", *Discrete Mathematics* 219 (2000), 207-219.

12. D.S. Archdeacon, C.J. Colbourn, I. Gitler, and J.S. Provan, "Four-terminal reducibility and projective-planar wye-delta-wye reducible graphs", *J. Graph Theory* 37 (2000), 83-93.

13. C.J. Colbourn and A.C.H. Ling, "Balanced sampling plans with block size four excluding contiguous units", *Australasian J. Combinatorics* 20 (1999), 37-46.

14. C.J. Colbourn, D.L. Kreher, J.P. McSorley, and D.R. Stinson, "Orthogonal arrays of strength three from regular 3–wise balanced designs", *J. Statistical Planning Inference*, to appear (acc May 99).

15. C.J. Colbourn, J.H. Dinitz, and A. Rosa, "Bicoloring Steiner triple systems", *Electronic J. Combinatorics* 6 (1999), #R25.

16. Y.M. Chee, C.J. Colbourn, and A.C.H. Ling, "Asymptotically optimal erasure-resilient codes for large disk arrays", *Discrete Applied Mathematics* 102 (2000), 3-36.

17. A.C.H. Ling, C.J. Colbourn, M.J. Grannell, and T.S. Griggs, "Construction techniques for anti-Pasch Steiner triple systems", *J. London Mathematical Society* (2) 61 (2000), 641-657.

18. C.J. Colbourn and J.H. Dinitz, "Mutually orthogonal latin squares: a brief survey of constructions", *J. Statistical Planning Inference* 95 (2001), 9–48.

19. C.J. Colbourn and S. Zhao, "Maximum Kirkman signal sets for synchronous uni-polar multi-user communication systems", *Designs Codes Cryptography* 20 (2000), 219-227.

20. C.J. Colbourn, M.A. Oravas, and R.S. Rees, "Steiner triple systems with disjoint or intersecting subsystems", *J. Combinatorial Designs* 8 (2000), 58-77.

21. C.J. Colbourn, "Group testing for consecutive positives", *Annals of Combinatorics* 3 (1999), 37–41.

22. M.A. Chateauneuf, C.J. Colbourn, D.L. Kreher, E.R. Lamken, and D.C. Torney, "Pooling, lattice square, and union jack designs", *Annals of Combinatorics* 3 (1999), 27–35.

23. C.J. Colbourn and J.H. Dinitz, "Generating sets in Steiner triple systems", *Math. Slovaca* 50 (2000), 259–269.

24. C.J. Colbourn, "Weakly union-free maximum packings", *Annals of Combinatorics* 3 (1999), 43–52.

25. C.J. Colbourn and A.C.H. Ling, "Kirkman school project designs", *Discrete Mathematics* 203 (1999), 49–60.

26. L. Riccio and C.J. Colbourn, "Sharper bounds in adaptive group testing", *Taiwanese J. Mathematics* 4 (2000), 669-673.

27. M.A. Chateauneuf, C.J. Colbourn, and D.L. Kreher, "Covering arrays of strength three", *Designs Codes Cryptography* 16 (1999), 235–242.

28. C.J. Colbourn, "A Steiner 2–design with an automorphism fixing exactly $r + 2$ points", *J. Combinatorial Designs* 7 (1999), 375–380.

29. C.J. Colbourn, "Minimum weights of point codes of Steiner triple systems", *J. Statistical Planning Inference* 95 (2001), 161–166.

30. M.K. Chari and C.J. Colbourn, "Reliability polynomials: A survey", *J. Combinatorics, Information and System Sciences* 22 (1997), 177-193.

31. C.J. Colbourn and A.C.H. Ling, "A class of partial triple systems with applications in survey sampling", *Communications in Statistics: Theory and Methods* 27 (1998), 1009–1018.

32. C.J. Colbourn and A.C.H. Ling, "Point code minimum Steiner triple systems", *Designs Codes Cryptography* 14 (1998), 141–146.

33. C.J. Colbourn and J.H. Dinitz, "Complete arcs in Steiner triple systems", *J. Combinatorial Theory (A)* 80 (1997), 320–333.

34. Y.M. Chee, C.J. Colbourn, and A.C.H. Ling, "Weakly union-free twofold triple systems", *Annals of Combinatorics* 1 (1997), 215–225.

35. C.J. Colbourn and G. Xue, "A linear time algorithm for computing the most reliable source on a series-parallel graph with unreliable edges", *Theoretical Computer Science* 209 (1998), 331–345.

36. M.B. Cohen, C.J. Colbourn and D. Froncek, Cluttered orderings for the complete graph, *COCOON 2001*, August 2001, proceedings to appear.

37. M.B. Cohen and C.J. Colbourn, Ordering disks for double erasure codes, *Proc. Symp. Parallel Algorithms and Architectures*, Crete, July 2001, to appear.

38. D.R. de la Torre, C.J. Colbourn, and A.C.H. Ling, "An application of permutation arrays to block ciphers", *Proceedings Thirty-first Southeastern Conference on Combinatorics, Graph Theory, and Computing, Congressus Numerantium* 145 (2000), 5-7.

39. M.B. Cohen and C.J. Colbourn, "Steiner triple systems as multiple erasure correcting codes in disk arrays", *Proceedings of IPCCC 2000 (19th IEEE International Conference on Performance, Computing and Communications)*, 2000, pp. 288-294.

40. M.B. Cohen and C.J. Colbourn, "Optimal and pessimal orderings of Steiner triple systems in disk arrays", LATIN 2000 (Punta del Este, Uruguay), *Lecture Notes in Computer Science* 1776 (2000), 95-104.

41. H.J. Strayer, K.C. Wellsch, C.J. Colbourn, and F. Glover, "Planarization, dualization, and all-terminal reliability", *Proceedings of the Eighth International Conference on Graph Theory, Combinatorics, Algorithms and Applications* (Y. Alavi, D.R. Lick, and A. Schwenk; editors) Volume II, New Issues Press, Kalamazoo MI, 1999, pp. 873-882.

42. C.J. Colbourn and A. Rosa, *Triple Systems*, Oxford University Press, Oxford, 1999. ISBN 0 19 853576 7.

43. C.J. Colbourn and G. Xue, "Grade of service Steiner trees in series-parallel networks", *Advances in Steiner Trees* (D.Z. Du, J.M. Smith, J.H. Rubinstein; eds.) Kluwer Academic Press, 2000, pp. 163–174.

44. chapters in the *CRC Handbook of Discrete and Combinatorial Mathematics* (K. Rosen *et al.*, editors), CRC Press, 2000:

Sect	Authors	Title	Pages
8.1	C.J. Colbourn, J.H. Dinitz	Block designs	754-770
8.2	C.J. Colbourn, J.H. Dinitz	Symmetric designs and finite geometries	770-778
8.3	C.J. Colbourn, J.H. Dinitz	Latin squares and orthogonal arrays	778-786

45. C.J. Colbourn, "Reliability issues in telecommunications network planning", in: *Telecommunications Network Planning* (B. Sansó, ed.) Kluwer Academic Press, 1999, pp. 135-146.

46. C.J. Colbourn, J.H. Dinitz and D.R. Stinson, "Applications of combinatorial designs to communications, cryptography, and networking", *Surveys in Combinatorics 1999* (J.D. Lamb and D.A. Preece, eds.), Cambridge University Press, pp. 37-100.

7. Participating Scientific Personnel

Myra Cohen: completed a Master's degree in Computer Science in 1999, and continued to work on the project. She is now a Ph.D. student at University of Auckland.

Xunshan Ma: completed a Master's degree in Computer Science in 1999.

Laura Riccio: was involved as an undergraduate and went on to pursue a Ph.D. at the University of California at Berkeley.

Lee Ives: was involved as an undergraduate.

Sufang Zhao: was a PDF on the project; she went on to an industry position.

Alan Ling: was a PDF on the project, now an assistant professor at Michigan Tech.

Feliú Sagols: was a PDF on the project, now an assistant professor in Mexico City.

Alex Rosa: was a short term visiting Professor working on the project.

Shen Hao: was a short term visiting Professor working on the project.

References

[1] N. Alon, J. Edmonds, and M. Luby. Linear time erasure codes with nearly optimal recovery. In *Proc. 36th Annual IEEE Symposium on Foundations of Computer Science*, pages 512–519. IEEE, 1995.

[2] G. M. Amdahl. Validity of the single processor approach to achieving large scale computing capabilities. In *Proc. AFIPS 1967 Spring Joint Computer Conference*, volume 30, pages 483–485, Washington, D. C., 1967. AFIPS.

[3] M. O. Ball, C. J. Colbourn, and J. S. Provan. Network reliability. In *Handbook of Operations Research: Network Models*, pages 673–762, Amsterdam, 1995. Elsevier North-Holland.

[4] M. Blaum, J. Brady, J. Bruck, and J. Menon, EVENODD: an efficient scheme for tolerating double disk failures in RAID architectures, *IEEE Transactions on Computers* 44 (1995), 192–202.

[5] M. Blaum and R. M. Roth, On lowest density MDS codes, *IEEE Trans. Information Theory* 45 (1999), 46–59.

[6] J. W. Byers, M. Luby, M. Mitzenmacher, and A. Rege, A Digital Fountain Approach to Reliable Distribution of Bulk Data, *Proc. SIGCOMM Conference*, 1998, 56–67.

[7] M.A. Chateauneuf, C.J. Colbourn, D.L. Kreher, E.R. Lamken, and D.C. Torney, Pooling, lattice square, and union jack designs, *Annals of Combinatorics* 3 (1999), 27–35.

[8] Y. M. Chee, C. J. Colbourn, and A. C. H. Ling. Asymptotically optimal erasure-resilient codes for large disk arrays. *Discrete Applied Mathematics* 102 (2000), 3-36.

[9] Y. M. Chee, C. J. Colbourn, and A. C. H. Ling. Weakly union-free twofold triple systems, *Annals Combinatorics* 1 (1997) 215–225.

[10] P. M. Chen, E. K. Lee, G. A. Gibson, R. H. Katz, and D. A. Patterson. RAID: High-performance, reliable secondary storage. *ACM Comput. Surveys*, 26:145–185, 1994.

[11] Myra B. Cohen. Performance analysis of triple erasure codes in large disk arrays. Master's thesis, University of Vermont, 1999.

[12] M. B. Cohen, Double erasure codes, invited lecture at the Workshop on Emerging Applications of Combinatorial Designs, Mathematical Sciences Research Institute, UC Berkeley, November 2000.

[13] M. B. Cohen and C. J. Colbourn, Steiner triple systems as multiple erasure correcting codes in disk arrays, *Proceedings of IPCCC 2000 (19th IEEE International Conference on Performance, Computing and Communications)* , 2000, pp. 288–294.

[14] M. B. Cohen and C. J. Colbourn, Optimal and pessimal orderings of Steiner triple systems in disk arrays, *Theoretical Computer Science*, in press, 2000. Preliminary version in: LATIN 2000 (Punta del Este, Uruguay), *Lecture Notes in Computer Science* 1776 (2000), 95-104.

[15] M.B. Cohen, C.J. Colbourn, L.A. Ives and A.C.H. Ling, Kirkman Triple Systems of Orders 21 with nontrivial automorphism group, submitted for publication.

[16] C. J. Colbourn, Weakly union-free maximum packings, *Annals Combinatorics* 3 (1999), 43–52.

[17] C. J. Colbourn, Projective planes and congestion-free networks, preprint, University of Vermont, 1998.

[18] C. J. Colbourn, Emerging applications of designs in communications and computing, keynote lecture at the Workshop on Emerging Applications of Combinatorial Designs, Mathematical Sciences Research Institute, UC Berkeley, November 2000.

[19] C. J. Colbourn and J. H. Dinitz (editors), *CRC Handbook of Combinatorial Designs* CRC Press, Boca Raton FL, 1996.

[20] C. J. Colbourn, J. H. Dinitz, and D. R. Stinson, Applications of combinatorial designs to communications, cryptography, and networking, *Surveys in Combinatorics 1999* (J.D. Lamb and D.A. Preece; eds.), Cambridge University Press, pp. 37-100.

[21] C.J. Colbourn, J.H. Dinitz, and D.R. Stinson, Quorum systems constructed from combinatorial designs, *Information and Computation*, in press.

[22] C. J. Colbourn and A. C. H. Ling, Quorums from difference covers, *Information Processing Letters*, in press.

[23] C. J. Colbourn and A. Rosa, *Triple Systems* Oxford University Press, Oxford, 1999.

[24] C.J. Colbourn and F. Sagols, Triangulations and a generalization of Bose's method, *Discrete Mathematics*, in press.

[25] C.J. Colbourn and P.J. Wan, Minimizing Drop Cost for SONET/WDM Networks with $\frac{1}{8}$ Wavelength Requirements, *Networks*, in press.

[26] C. J. Colbourn and S. Zhao, Maximum Kirkman signal sets for synchronous uni-polar multi-user communication systems, *Designs, Codes and Cryptography* 20 (2000), 219–227.

[27] W.V. Courtright II., G. Gibson, M. Holland, L.N. Reilly and J. Zelenka. *RAIDframe: A Rapid Prototyping Tool for RAID Systems* Carnegie Mellon University, Pittsburgh, Pennsylvania, 1996.

[28] P. Elias, Coding for two noisy channels, in: *Proc. Third London Symp. Information Theory*, Butterworths, London, 1955, pp. 61–76.

[29] G. A. Gibson. *Redundant Disk Arrays: Reliable, Parallel Secondary Storage.* MIT Press, Cambridge, Massachusetts, 1992.

[30] M.J. Grannell, T.S. Griggs, and C.A. Whitehead. The resolution of the anti-Pasch conjecture. *Journal of Combinatorial Designs* 8 (2000), 300–309.

[31] D. D. Harms, M. Kraetzl, C. J. Colbourn, and J. S. Devitt. *Network Reliability: Experiments with a Symbolic Algebra Environment.* CRC Press, Boca Raton, Florida, 1995.

[32] L. Hellerstein, G. A. Gibson, R. M. Karp, R. H. Katz, and D. A. Patterson. Coding techniques for handling failures in large disk arrays. *Algorithmica*, 12:182–208, 1994.

[33] M. C. Holland. *On-Line Data Reconstruction In Redundant Disk Arrays.* PhD thesis, Carnegie Mellon University, 1994.

[34] E. K. Lee. Software and performance issues in the implementation of a RAID prototype. Tech. Rep. CSD-90-573, University of California at Berkeley, 1990.

[35] H. Lefmann, K. T. Phelps, and V. Rödl, On sparse parity check matrices, *Designs Codes and Cryptography* 12 (1997), 107–130.

[36] A.C.H. Ling, C.J. Colbourn, M.J. Grannell and T.S. Griggs. Construction techniques for anti-Pasch Steiner triple systems. *Journal of the London Mathematical Society*, in press.

[37] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, and D. A. Spielman, Analysis of Low Density Codes and Improved Designs Using Irregular Graphs, *Proc. ACM Symp. Theory of Computing*, 1998.

[38] M. Luby, M. Mitzenmacher, M. A. Shokrollahi, D. A. Spielman, and V. Stemann, Practical loss–resilient codes, *Proc. ACM Symp. Theory of Computing*, 1997.

[39] Xunshan Ma, Computational method to construct erasure-resilient codes, Master's thesis, Computer Science, University of Vermont, 1999.

[40] F. J. MacWilliams and N. J. A. Sloane, *The Theory of Error-Correcting Codes*, North-Holland, Amsterdam, 1978.

[41] Paul Massiglia. *The RAID Book, A Storage System Technology Handbook, 6th Edition.* The RAID Advisory Board, 1997.

[42] D. A. Patterson and J. L. Hennessy. *Computer Organization and Design: The Hardware/Software Interface.* Morgan Kaufmann, San Mateo, California, 1994.

[43] M. O. Rabin. Efficient dispersal of information for security, load balancing, and fault tolerance. *J. Assoc. Comput. Mach.*, 36:335–348, 1989.

[44] F. Sagols and C.J. Colbourn, NS1D0 Sequences and Anti-Pasch Steiner Triple Systems, *Ars Combinatoria*, in press.

[45] K. Salem and H. Garcia-Molina. Disk striping. In *Proc. 2nd International Conference on Data Engineering*, pages 336–342, Washington, D. C., 1986. IEEE.

[46] M. A. Shokrollahi, New Sequences of Linear Time Erasure Codes Approaching the Channel Capacity, *Lecture Notes in Computer Science* 1719 (1999), 65–76.

[47] D. R. Stinson, Tran van Trung and R. Wei, Secure frameproof codes, key distribution patterns, group testing algorithms, and related structures, *Journal of Statistical Planning and Inference* 86 (2000), 595–617.

[48] D. R. Stinson and R. Wei, Combinatorial properties and constructions of traceability schemes and frameproof codes, *SIAM Journal of Discrete Mathematics* 11 (1998) 41–53.

[49] L. Xu, V. Bohossian, J. Bruck, D. G. Wagner, Low density MDS codes and factors of complete graphs, *IEEE Trans. Information Theory* 45 (1999), 1817-1826.

[50] S. Zhao, Application of BIBDs in MT-MFSK signal set design for multiplexing bursty sources, PhD thesis, University of Technology Sydney, 1998.

[51] S. Zhao, K. W. Yates and K. Yasukawa, Application of Kirkman designs in joint detection multiple access schemes, *Proceedings of the International Symposium on Spread Spectrum Techniques an d Applications* 2 (1996) 857–861.